AMENDMENTS TO THE CLAIMS

This listing of claims replaces all previous versions and listings of claims in this application.

Claim Listing:

Claims 1-6: (Cancelled).

7. (Currently amended) In a multiple computer system including a plurality of single computers interconnected via a communications link, a method of loading an application program onto each of said plurality of single computers, the application program having application program code including a plurality of code threads all intended to execute on and reference a single computer having a single processing unit or symmetric multiple processing units and a single independent local memory with a local memory capacity that is not shared with any other single computer of said plurality of single computers, the method comprising ~~the steps of~~:

loading the application program written to operate only on a single computer onto each different computer of said plurality of single computers;

modifying the application program on each said different single computer before execution of said corresponding portion of the application program written to operate only on a single computer on each said different single computer;

~~substantially~~ simultaneously executing different portions of said application program on each different one of the plurality of single computers with each different one of the plurality of single computers having a different independent local memory accessible only by a corresponding portion of the application program; and

restricting read requests of each and every said computer such that all read requests of local memory of the requesting computer and not reading from the memory of any other computer;

wherein the step of modifying comprises:

(i) detecting instructions in the unmodified application program which reference the same

common memory records;

(ii) listing all such commonly referenced memory records by a distributed runtime (DRT) and providing a naming tag for each said listed commonly referenced memory record;

(iii) detecting those instructions which write to, or manipulate the contents of, any of said listed commonly referenced memory records; and

(iv) generating and inserting an alert instruction into the unmodified application program to create the modified application program for handling by the DRT following each said detected commonly referenced memory record write or manipulate instruction indicating that the contents or value of the commonly referenced memory record were re-written or manipulated and may have changed during execution of a code thread, and

wherein:

said alert instruction being operative for initiating propagation of the re-written or manipulated contents and name tag of each said re-written or manipulated listed commonly referenced memory record via the communications link to the distributed run times (DRTs) of each other of the single computers;

each DRT creates a table when initially recording fields, and for each field there is a name which is common throughout the network and which the network recognizes; and

wherein in different ones of said plurality of single computers, a memory location corresponding to a given name field will vary over time and each of the DRTs will have different memory locations but each field name will have the same field value stored in the different memory locations, and wherein the DRT initially creates a JAVA program language byte code virtual machine for execution of the modified application program; and

different portions of said modified application program being simultaneously executable on each different one of the plurality of single computers with each different one of the plurality of single computers having a different independent local memory accessible only by a corresponding portion of the application program.

8. (Previously Presented) The method as claimed in claim 7 wherein the step of

modifying the application program is different for different computers.

Claims 9-14: (Cancelled).

15. (Previously Presented) The method as claimed in claim 31, and carried out prior to loading the application program onto each said computer.

16. (Previously Presented) The method as claimed in claim 31, and carried out during loading of the application program onto each said computer.

17. (Previously Presented) The method as claimed in claim 31, and carried out by just-in-time compilation.

18. (Previously Presented) The method as claimed in claim 31, and carried out by re-compilation after loading.

Claims 19-23 (Cancelled)

24. (Previously Presented) A method of loading an application program as in claim 7, wherein said program written to operate on only a single computer is a program written to execute within a local processor or processors and local memory coupled to the processor or processors within the single computer.

25. (Previously Presented) A method of loading an application program as in claim 7, wherein each of the computers operates with the same application program and data and thus all of the plurality of computers have the same application program and data.

26. (Previously Presented) A method of compiling or modifying an application program as in claim 31, wherein said program written to operate on only a single computer is a program written to execute within a local processor or processors and local memory coupled to the processor or processors within the single computer.

Claims 17-28 (Cancelled)

29. (Previously Presented) The method as claimed in claim 7, wherein the interconnection of the plurality of computers via the communications link without forming a distributed shared memory arrangement and the different portions of said application program

being simultaneously executable on each different one of the plurality of computers with each different one of the plurality of computers having a different independent local memory accessible only by a corresponding portion of the application program eliminate clock cycle delays that would otherwise be associated with one or said plurality of computers reading memory physically located in a different one or ones of the plurality of computers formed in a distributed shared memory arrangement.

30. (Previously Presented) The method as claimed in claim 25, wherein the interconnection of the plurality of computers via the communications link without forming a distributed shared memory arrangement and the different portions of said application program being simultaneously executable on each different one of the plurality of computers with each different one of the plurality of computers having a different independent local memory accessible only by a corresponding portion of the application program eliminate clock cycle delays that would otherwise be associated with one or said plurality of computers reading memory physically located in a different one or ones of the plurality of computers formed in a distributed shared memory arrangement.

31. (Currently amended) A method of compiling or modifying an application program written to include a plurality of instruction code threads intended to execute on and reference only a single computer having a single central processing unit (CPU) or symmetric multiple processing units and a single independent local memory that is not shared with any other computer of a plurality of single computers but said application program to run simultaneously on each one of said plurality of single computers interconnectable via a communications link, with different portions of said application program being simultaneously executable on different ones of said plurality of single computers with each one of the plurality of single computers having the independent local memory accessible only by the corresponding portion of the application program, said method comprising ~~the steps of~~:

(i) detecting instructions in the unmodified application program which reference the same common memory records;

(ii) listing all such commonly referenced memory records and providing a naming tag for each said listed commonly referenced memory record;

(iii) detecting those instructions which write to, or manipulate the contents of, any of said listed commonly referenced memory records;

(iv) generating and inserting an alert instruction into the unmodified application program to create the modified application program for handling by a distributed run time (DRT) following each said detected commonly referenced memory record write or manipulate instruction indicating that the contents or value of the commonly referenced memory record were re-written or manipulated and may have changed during execution of a code thread, said alert instruction being operative for initiating propagation of the re-written or manipulated contents and name tag of each said re-written or manipulated listed commonly referenced memory record via the communications link to the distributed run times (DRTs) of each other of the single computers; and

restricting read requests of each and every said computer such that all read requests of each and every said computer are satisfied by reading only a corresponding independent local memory of the requesting computer and not reading from the memory of any other computer,

wherein:

step (ii) includes listing all such commonly referenced memory records by a distributed runtime (DRT) and providing a naming tag for each said listed commonly referenced memory record; and

each DRT creates a table when initially recording fields, and for each field there is a name which is common throughout the network and which the network recognizes;

wherein, in different ones of said plurality of single computers, a memory location corresponding to a given name field will vary over time and each of the DRTs will have different memory locations but each field name will have the same field value stored in the different memory locations; and

the DRT initially creates a virtual machine for execution of the modified application program.

32. (Previously Presented) A method as in claim 31, wherein the code thread that alerts the DRT to the re-writing or manipulation and possible change of contents or value of the

commonly referenced memory record also performs at least one of:

(i) directly notify and propagate to all other DRTs executing on each other one of the plurality of single computers of the re-writing or manipulation and possible change of contents or value of the commonly referenced memory record and then resumes processing; and

(ii) indirectly notify and propagate by instructing another thread to notify and propagate the all other DRTs executing on each other one of the plurality of single computers of the re-writing or manipulation and possible change of contents or value of the commonly referenced memory record and then resumes processing.

33. (Currently amended) A method as in claim 32, wherein when the notification and propagation are indirect, the processing of ~~the alerted thread~~ code thread that alerts the DRT is only interrupted momentarily before the alerted thread processing resumes and said another thread which has been notified of the re-written or manipulated commonly referenced memory record then communicates that re-written or manipulated commonly referenced memory record to each of the other single computers so that better utilization of the processing power of various executing threads and gives better scaling with increasing number of single computers when the application program is executed.

34. (Previously presented) A method as in claim 31, wherein the communication link comprises the Internet.

35. (Previously Presented) A method as in claim 31, wherein the communication link comprises an intranet.

36. (Previously Presented) A method as in claim 3 1, wherein the communication link comprises a local area network.

37. (Previously Presented) A method as in claim 31, wherein the commonly referenced memory locations comprise JAVA programming language fields and the contents or values stored in the commonly referenced memory locations comprise JAVA programming language field contents or values.

38. (Previously Presented) A method as in claim 31, wherein the commonly referenced memory records comprise JAVA programming language fields and the JAVA

programming language fields are listed by object and class.

39. (Previously Presented) A method as in claim 31, wherein the application program is written in the JAVA programming language and the step of detecting instructions in the unmodified application program which reference the same common memory records comprise searching through the JAVA programming language code and identifying a put static (putstatic) instruction and generating and inserting an alert instruction into the JAVA application program for each said putstatic instruction so identified.

40. (Previously Presented) A method as in claim 39, further comprising:

modifying the JAVA application program so that during execution of the modified JAVA application program upon executing the inserted alert instruction notification, sending the commonly referenced memory record that was re-written or manipulated and may have changed during execution of a code thread with its name tag across the network and receiving the commonly referenced memory record that was re-written or manipulated and may have changed during execution of a code thread with its name tag by a different computer.

41. (Previously Presented) A method as in claim 31, wherein a multicast socket is used for a distributed run time (DRT) communication of the commonly referenced memory record that was re-written or manipulated and may have changed during execution of a code thread with its name tag.

42. (Previously Presented) A method as in claim 31, wherein the updating of all of the commonly referenced memory records that were re-written or manipulated and may have changed during execution of code threads are updated over the Internet.

43. (Previously Presented) A method as in claim 31, wherein the communication link comprises the Internet and all updates to commonly referenced memory locations are performed using Internet network packets through separate distributed runtimes (DRTs) executing on each of the plurality of single computers.

44. (Previously Presented) A method as in claim 31, further comprising writing the value from the network packet for the commonly referenced memory record that was rewritten or manipulated and may have changed into the memory location of the receiving computer.

45. (Currently amended) In a multiple computer system including a plurality of single computers interconnectable via an Internet or intranet network communications link, a method of loading an original application program onto each of said plurality of single computers, the original application program having original application program code including a plurality of original code threads all written to execute on and reference a single computer having a single processing unit or symmetric multiple processing units and a single local memory with a local memory capacity that is not shared with any other single computer of said plurality of single computers, the system configured to enable simultaneous cooperative execution of said application program by said plurality of single computers, with the original application program being modified to form at least one modified application program with different portions of said modified application program being simultaneously executed within a different independent local processor and a different independent local memory within each different one of the plurality of single computers, said different independent local memory within each said different single computer not forming a distributed shared memory arrangement and being accessible during execution of said application program and said different portions of said application program only by the different portion of the application program actually executing within the different local processing unit or symmetric multiple processing units of the different computer, the method comprising ~~the steps of~~:

loading the application program onto each different computer of said plurality of single computers, said application program including a reference to a program memory field that may be referenced by one or more of said plurality of computers during execution of their respective different portion of the application program; and

modifying the application program on each said different single computer before execution of said different portion of the application program on each said different single computer; and

restricting read requests of each and every said computer such that all read requests of each and every said computer are satisfied by reading only the corresponding independent local memory of the requesting computer and not reading from the memory of any other computer; and

wherein said modification of the application program includes an insertion of at least one

code thread prior to execution that upon execution by one of said single computers initiates a sequence of events that result in a network packet communication over said Internet or intranet network communications link that contains an identifier of the referenced memory field and the contents or value of that memory field,

wherein said modifying comprises:

(i) detecting instructions in the unmodified application program which reference the same common memory records;

(ii) listing all such commonly referenced memory records by a distributed runtime (DRT) and providing a naming tag for each said listed commonly referenced memory record;

(iii) detecting those instructions which write to, or manipulate the contents of, any of said listed commonly referenced memory records; and

(iv) generating and inserting an alert instruction into the unmodified application program to create the modified application program for handling by the DRT following each said detected commonly referenced memory record write or manipulate instruction indicating that the contents or value of the commonly referenced memory record were re-written or manipulated and may have changed during execution of a code thread, and

wherein:

said alert instruction are operative for initiating propagation of the re-written or manipulated contents and name tag of each said re-written or manipulated listed commonly referenced memory record via the communications link to the distributed run times (DRTs) of each other of the single computers;

each DRT creates a table when initially recording fields, and for each field there is a name which is common throughout the network and which the network recognizes;

wherein, in different ones of said plurality of single computers, a memory location corresponding to a given name field will vary over time and each of the DRTs will have different memory locations but each field name will have the same field value stored in the different memory locations, and wherein the DRT initially creates a JAVA program language byte code

virtual machine for execution of the modified application program; and

different portions of said modified application program being simultaneously executable on each different one of the plurality of single computers with each different one of the plurality of single computers having a different independent local memory accessible only by a corresponding portion of the application program.

46. (Previously Presented) A method as in claim 45, further comprising executing said modified application program and generating and communicating said network packet communication over said Internet or intranet network communications link that contains said identifier of the referenced memory field and the contents or value of said referenced memory field.

Claim 47: (Cancelled).

48. (Previously Presented) The method as claimed in claim 7, wherein the step of modifying comprises:

(i) detecting instructions in the unmodified application program which reference the same common memory records;

(ii) listing all such commonly referenced memory records by a distributed runtime (DRT) and providing a naming tag for each said listed commonly referenced memory record;

(iii) detecting those instructions which write to, or manipulate the contents of, any of said listed commonly referenced memory records; and

(iv) generating and inserting an alert instruction into the unmodified application program to create the modified application program for handling by the DRT following each said detected commonly referenced memory record write or manipulate instruction indicating that the contents or value of the commonly referenced memory record were re-written or manipulated and may have changed during execution of a code thread.

49. (Currently amended) The method as claimed in claim 48, wherein:

said alert instruction being operative for initiating propagation of the re-written or

manipulated contents and name tag of each said re-written or manipulated listed commonly referenced memory record via the communications link to the distributed run times (DRTs) of each other of the single computers;

each DRT creates a table when initially recording fields, and for each field there is a name which is common throughout the network and which the network recognizes; and

wherein in different ones of said plurality of single computers, a memory location corresponding to a given name field will vary over time and each of the DRTs will have different memory locations but each field name will have the same field value stored in the different memory locations, and wherein the DRT initially creates a JAVA program language byte code virtual machine for execution of the modified application program; and

different portions of said modified application program being ~~substantially~~ simultaneously executable on each different one of the plurality of single computers with each different one of the plurality of single computers having a different independent local memory accessible only by a corresponding portion of the application program.

Claims 50-52: (Canceled).

52. (Previously Presented) The method as claimed in claim 31, wherein:

step (ii) listing all such commonly referenced memory records and providing a naming tag for each said listed commonly referenced memory record includes listing all such commonly referenced memory records by a distributed runtime (DRT) and providing a naming tag for each said listed commonly referenced memory record; and

each DRT creates a table when initially recording fields, and for each field there is a name which is common throughout the network and which the network recognizes;

wherein in different ones of said plurality of single computers, a memory location corresponding to a given name field will vary over time and each of the DRTs will have different memory locations but each field name will have the same field value stored in the different memory locations; and

the DRT initially creates a virtual machine for execution of the modified application

program.

53. (Currently amended) The method as claimed in ~~claim 52,~~ claim 31, wherein the DRT created virtual machine comprises a JAVA programming language byte code virtual machine for execution of the modified application program.